an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution such that the number of said plurality of programs that are interleaved is greater than or equal to the depth of the pipeline.

2. (ONCE AMENDED) The processor of claim 1 or claim 23 wherein said pipeline has a datapath with a depth equal to said number of programs.

3. (ONCE AMENDED) The processor of claim 1 wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed.

4. (ONCE AMENDED) The processor of claim 1 or claim 23 wherein each program of said plurality of programs is independent of the other of said plurality of programs.

5. (ONCE AMENDED) The processor of claim 1 or claim 23 further including an output buffer for storing out of order data output.

6. (ONCE AMENDED) The processor of claim 1 or claim 23 further including one or more of a register copy, program counter, and program counter stack provided for each of said plurality of programs.

7. (ONCE AMENDED) The processor of claim 6 wherein one or more of control and computing resources, instructions, instruction memory, data paths, data memory, and caches are shared by said plurality of programs.

8. (ONCE AMENDED) The processor of claim 1 or claim 23 wherein said processor executes SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N.

9. (ONCE AMENDED) The processor of claim 1 or claim 23 wherein said instructions comprise load instructions for loading data from a data memory.

10. (ONCE AMENDED) The processor of claim 1 or claim 23 wherein said instructions comprise store instructions for storing data in a memory.

11. (ONCE AMENDED) The processor of claim 9 wherein said data memory comprises a cache.

12. (ONCE AMENDED) The processor of claim 9 wherein address space of said data memory comprises a frame buffer unit.

13. (ONCE AMENDED) The processor of claim 9 wherein address space of said data memory comprises a texture memory unit.

14. (ONCE AMENDED) A method of executing instructions from a plurality of programs comprising:

identifying N programs of said plurality of programs;

interleaving instructions from said N programs in a processor pipeline; and

executing said instructions such that a first instruction from one of said N programs is completed before beginning execution of a second instruction of said one of said N programs wherein no no-op is inserted into the pipeline for the purpose of ensuring that said first instruction is completed before beginning execution of said second instruction.

15. (ONCE AMENDED) The method of claim 14 or claim 24 further including the step of assigning a program counter to each of said N programs.

16. (ONCE AMENDED) The method of claim 14 or claim 24 further including the step of assigning a register to each of said N programs.

17. (ONCE AMENDED) The method of claim 14 or claim 24 wherein said graphics processing execution pipeline has a depth of N.

18. (ONCE AMENDED) The method of claim 14 or claim 24 wherein said processor executes SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N.

Please cancel claims 19-22.

Please add the following claims:

23. (NEW) A programmable processor for executing a plurality of programs, said programmable processor comprising:

an execution pipeline having a pipeline latency; and

an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed and wherein no no-op is inserted into the pipeline for the purpose of ensuring that said next instruction is not provided to said pipeline until said previous instruction has completed.

24. (NEW) A method of executing instructions from a plurality of programs comprising:

identifying N programs of said plurality of programs wherein N is greater than or equal to the depth of a processor pipeline;

interleaving instructions from said N programs in said processor pipeline; and

executing said instructions.

Clean Set of Claims

1. A programmable processor for executing a plurality of programs, said programmable processor comprising:

an execution pipeline having a pipeline latency; and

an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution such that the number of said plurality of programs that are interleaved is greater than or equal to the depth of the pipeline.

2. The processor of claim 1 or claim 23 wherein said pipeline has a datapath with a depth equal to said number of programs.

3. The processor of claim 1 wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed.

4. The processor of claim 1 or claim 23 wherein each program of said plurality of programs is independent of the other of said plurality of programs.

5. The processor of claim 1 or claim 23 further including an output buffer for storing out of order data output.

6. The processor of claim 1 or claim 23 further including one or more of a register copy, program counter, and program counter stack provided for each of said plurality of programs.

7. The processor of claim 6 wherein one or more of control and computing resources, instructions, instruction memory, data paths, data memory, and caches are shared by said plurality of programs.

8. The processor of claim 1 or claim 23 wherein said processor executes SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N.

9. The processor of claim 1 or claim 23 wherein said instructions comprise load instructions for loading data from a data memory.

10. The processor of claim 1 or claim 23 wherein said instructions comprise store instructions for storing data in a memory.

11. The processor of claim 9 wherein said data memory comprises a cache.

12. The processor of claim 9 wherein address space of said data memory comprises a frame buffer unit.

13. The processor of claim 9 wherein address space of said data memory comprises a texture memory unit.

14. A method of executing instructions from a plurality of programs comprising:

identifying N programs of said plurality of programs;

interleaving instructions from said N programs in a processor pipeline; and

executing said instructions such that a first instruction from one of said N programs is completed before beginning execution of a second instruction of said one of said N programs

wherein no no-op is inserted into the pipeline for the purpose of ensuring that said first instruction is completed before beginning execution of said second instruction.

15. The method of claim 14 or claim 24 further including the step of assigning a program counter to each of said N programs.

16. The method of claim 14 or claim 24 further including the step of assigning a register to each of said N programs.

17. The method of claim 14 or claim 24 wherein said graphics processing execution pipeline has a depth of N.

18. The method of claim 14 or claim 24 wherein said processor executes SIMD vector instructions of vector length N and executes in parallel a plurality of instructions having SIMD vector lengths that sum up to N.

23. A programmable processor for executing a plurality of programs, said programmable processor comprising:

an execution pipeline having a pipeline latency; and

an interleaver for interleaving instructions from said plurality of programs and providing said instructions to said pipeline for execution wherein a next instruction from one of said plurality of programs is not provided to said pipeline until a previous instruction of said one of said plurality of programs has completed and wherein no no-op is inserted into the pipeline for the purpose of ensuring that said next instruction is not provided to said pipeline until said previous instruction has completed.

24. A method of executing instructions from a plurality of programs comprising:

identifying N programs of said plurality of programs wherein N is greater than or equal to the depth of a processor pipeline;

interleaving instructions from said N programs in said processor pipeline; and

executing said instructions.